AMIGA WORKBENCH

FOR THE COMMODORE AMIGA USER

Volume 1, Issue 5 Circulation: 300 October 1986

Next Meeting

Sunday, October 12th at 2pm

Theme: Amiga Music

AUG meetings are held at Victoria College, Burwood Campus, in Lecture Theatre 2 and building E. Melways map 61 reference B5.

Amiga User's Group, PO Box 109, North Balwyn, 3104, Victoria, Australia

AMIGA™ Users Group

P.O. Box 109, North Balwyn, Victoria, 3104

Co-Ordinator's Report

Now that we've had a couple of General Meetings and Committee Meetings behind us and I think it's time to report some of the proceedings.

The General Meeting with Tony Cuffe was successful and Tony provided a wealth of information, as we had expected. I regret that I was away ski-ing at the time and missed it. We hope to have a variety of speakers and events in the future so that every meeting will be as informative and interesting as possible. We don't yet have a full agenda and suggestions are most welcome.

As to the Committee Meetings, some matters have arisen (and mostly been resolved) which need to be passed on to you.

MONEY: The influx of money has been very fast, almost too fast to be counted! This is gratifying, and suggests that the Group will be financially sound enough to achieve many of the things we hope for and are planning. I think you should know that positive steps are being taken to ensure that proper accounting procedures are followed, receipts will be issued for all monies taken, and the various divisions (such as Software, Purchasing, Membership) will account to the Treasurer for all Income and Expenditure. I can assure you that the Committee believes that the handling of Members' money (your money) is a serious business and will be taken seriously.

NEWSLETTER: Like all Publications, contributions are eagerly sought. I won't bore you with all the usual pleas since I know they're not necessary; articles will come flooding in. If they don't, I will FILL the magazine with all manner of pleas and whinges, thereby boring you to death.

SOFTWARE LIBRARY: One of our great success stories. Demand is enormous. So much so that we are trying NOT to copy AUG discs at meetings. There are two reasons; 1) someone has to do it, which means they miss out on the meeting, and 2) we feel the time could be better spent. We would prefer that orders are placed BEFORE the meeting (that means a week or two before, not five minutes before) and the discs picked up at the meeting, OR you can place an order at the meeting (with money paid then) and your discs will be posted out two weeks later. Naturally, for those people who cannot organise themselves this way, there will be some duplicating still carried out at meetings. This will probably be limited and some people may miss out due to time.

MAGAZINES: A list of articles is being compiled and both the list and the magazines will be available for perusal at meetings. One of the problems addressed by the Committee is the possibility that people borrowing magazines may not be able to return them until the next meeting. This is not satisfactory because it limits and slows circulation. We are trying to remedy this and will come up with something Real Soon Now.

PURCHASING: The Purchasing Officer ("it's-a-jungle-out-there, boy-have-I-got-a-deal-for-you, get-em-while-they're-hot") has been twisting arms all over town. This month's hot item is blank discs. If there is anything you want, let 'Drac' know, he'll most likely have a deal for you.

MEMBERSHIP: We currently in excess of TWO HUNDRED Members. Well done.

INCORPORATION: At the last General meeting, we voted to incorporate the Amiga Users Group, with our Secretary, Eric Salter, as the Public Officer. If you weren't at the last meeting, corner a committee member if you have any questions about Incorporation.

POLICY: Various policies are being evolved but one needs stating outright. Copying of Commercial Software is Piracy. It is not only against the best interests of the Club, it is illegal. We would not presume to tell people how to act elsewhere, but anyone found copying Commercial (Copyright) Software at Club Meetings will forthwith and immediately expelled from the Club and the media blanked.

ON A LIGHTER NOTE: Refreshments will be provided at future meetings, nothing elaborate, just coffee, soft drinks and stuff; feel free to bring some biscuits or cake, or even to help make the tea.

INPUT: Members who wish to attend Committee Meetings as observers are welcome to do so. Any information, suggestions, questions or (shudder) criticism can be sent to the Club's postal address. I encourage you to do this, it doesn't even need to be a proper letter. Do, however, write your name somewhere- ANON's will be ignored.

ATTENTION SPAN: This Report is the longest I'll ever write, it's just that there is a bit to say this time.

Happy crunching,

John Hollands

Production Credits

This month's **AMIGA WORKBENCH** was edited by Peter Jetson. Equipment and software used was: TurboDOS S-100 computer, Diablo 630 printer, Gemini 10x printer, Wordstar and Fancy Font.

Copyright and Reprint Privileges

Contributions

Articles, papers, letters, drawings and cartoons are actively sought for publication in **AMIGA WORKBENCH**. It would be appreciated if contributions were submitted on disk, since that means they don't have to be re-typed! We have access to a wide range of computers, so we should be able to accept almost any type of disk, but AMIGA disks are certainly the easiest. Absolute deadline for articles is the last weekend of the month before the cover date. Contributions can be sent to:

The Editor, AUG, PO Box 109, North Balwyn, 3104

The short item on setting the clock from the startup sequence in the July Amiga Workbench and a bad experience with an A-Time module that didn't like my printer got me to thinking (anyway thats my excuse). There I was with a real time (almost) clock in my micro for the first time and I wasn't using it because setting it every boot was too much of a hassle. I had to do something about that.

Most of my time is spent working on mainframe computers. They always have the clock set and tell me all sorts of interesting(?) things every time I use them like when I last logged on etc. Well if an IBM 3084Q can do it why can't my Amiga? The answer, of course, is that it can and now does.

The startup sequence below gives the last time the disk was booted, prompts for the time and date, updates a usage log and then invokes AmigaBASIC. If the files last-session or usage-log don't exist they are created.

There is still room for improvement of course, error trapping on user input is an obvious one and it can be easily modified to invoke any application or use a battery backed clock if you have one.

This is a fairly trivial application for a machine like an Amiga but it was an interesting way to learn more about CLI commands and the Amiga itself. It also has the advantage of doing something useful by getting the clock set and keeping track of how many times I have booted a particular disk.

If any of you have interesting little command lists or programs like this I'd love to see them, it's amazing what you can learn about a machine from something as small and simple as this.

Startup sequence to set clock and log disk usage

```
echo " "
echo "AmigaBASIC work disk.  Release 1.1"
echo " "
if exists last-session
     echo "Last session started at -"
     type last-session
else
     echo "No previous session logged"
endif
echo " "
echo "Enter current date and time"
echo "DD-MMM-YY HH:MM:SS"
echo " "
date >nil: ?
date >last-session
echo " "
if exists usage-log
     join usage-log last-session as ram:log
     copy ram:log to usage-log quiet
     delete ram:log
else
     copy last-session to usage-log
endif
echo "New session started at -"
date
wait 5
AmigaBASIC
```

-- From Bernd Kuenne.

Compute!'s Amiga Programmers Guide, a review

This book is edited by Stephen Levy and costs $38.95. It is a book of 458 pages divided into eight chapters covering the Amiga itself, AmigaBASIC, AmigaDOS, C, machine language programming, graphics and sound.

Chapter 1 covers the Amiga hardware itself and such things as windows, icons and gadgets at an introductory level. It also has a good go at being a sales pitch with glowing praise for the Amigas abilities. This chapter will leave you either with the warm glow of a good decision vindicated or the nagging feeling you got a sales glossy instead of a reference book for your money.

Chapter 2 covers Microsoft AmigaBASIC. It appears to be largely a rework of part of the information available in chapter 8 of the AmigaBASIC manual that came with my 1.1 upgrade. To be fair the explanations and examples have been reworded and the ones I checked appear easy to follow and generally in keeping with the introductory level of the book.

Chapter 3 introduces AmigaDOS and the CLI. It explains how to bring the CLI up, the basics of file names and path names and then covers the basic AmigaDOS commands. Again it is all at an introductory level. Most of this material is not available in the documentation that comes with the Amiga and would be useful to anyone who does not have a copy of the AmigaDOS User's Manual or wants something easier to get started with. Appendix B provides a more detailed description of AmigaDOS commands.

Chapter 4 covers command files (batch files to the author and those with IBM PC backgrounds) and Ed. All this in nine pages! This is also information missing from the standard documentation. Again appendix B should be used for detailed descriptions of the commands.

Chapter 5 deals with graphics and is much more substantial. The subject is presented in the context of AmigaBASIC and all examples are given in this language. Sprites, vsprites and bobs, the graphics library and how to call its functions from AmigaBASIC are all introduced in this chapter. Appendix G gives details of six graphics routines. Personally I think this is the most useful chapter in the book with chapter 6 a close second.

Chapter 6 is about speech and sound. Again this is a fairly substantial chapter with a number of example programs in AmigaBASIC. It also touches on machine language and sound giving an example assembler program, interesting stuff.

Chapter 7 is on the C programming language. I don't know C so I can't comment on the information given but it does have a good number of examples and seems to be at a fairly introductory level to me. It would probably help explain some of the Amiga specific questions that a general text on C would leave unanswered.

Chapter 8 is on machine language programming. It is fairly brief and doesn't do much more than give very high level overview of the subject. Interesting, but if you want to learn assembler programming you will need more than this book gives.

I have tried a number of the AmigaBASIC example programs and found them to be interesting demonstrations of the power of small BASIC programs on the Amiga. People new to micro's will be pleasantly surprised by what a few lines of BASIC can do.

I haven't tried any of the C or assembler programs but, going by what I have tried, I wouldn't expect them to give any trouble. The main C example is yet another mandelbrot program and is fairly large for this book with 22 pages of source code.

Looking back at what I have written it seems critical of the book and to be fair I think that I should say that the biggest fault of the book, to me at least, is that it tries to cover too much ground. The amount of information that a programmer might want about an Amiga is too large for one book and this one tries to cover most of it.

In summary this book would be useful to people with limited experience who want an easy introduction to their Amiga but it is not for those with more experience.

-- Bernd Kuenne

How To Run Public Domain Programs On A Single Drive Amiga by Bob Laidlaw

If you are very new to the Amiga and AmigaDos, you may have found that owning a single drive machine can be a source of frustration when trying to run the Public Domain Amiga Library disks. It is not much help to be told to set up a "RamDisk" when you are a raw beginner with AmigaDos.

Tom at the Computer Department of the Technical Book and Magazine Company (295 Swanston Street, Melbourne, phone (03) 663 3951) was very helpful by providing me with a copy of his Ramdisk utility (called CLI:) for the price of a diskette. After several phone calls to Tom and to Peter Jetson, I have got the system going satisfactorily with their help. It can be recommended as a good way to get started, and the process will introduce you to some useful AmigaDos commands.

The purpose of this article is to share this information with other beginners like myself. It assumes that you have obtained this CLI: disk. To avoid confusion, I suggest you make sure that each step works on your Amiga before going on to the next step.

1. **Back-up the CLI: Disk**

Essential - you are going to do some work on this copy!

2. **Load the CLI: Disk**

After using Kickstart, insert the CLI: disk. You will see some information about the disk and its directory scroll up. To stop the scrolling, press the Spacebar. To restart scrolling , just press the backspace key.

This is followed by the contents of a Help text file (called Help.doc), ending with the CLI prompt 1>. There is no need to make notes if you have a suitable printer, because we will be printing this file at Step 6 below.

At this point, I suggest that you run the following commands on your CLI: disk to give you some practice and confidence in viewing and printing directories and text files.

3. **View the CLI: Directory**

Type: **dir <RETURN>**

The directory listing will scroll up.

4. **Set the Preferences**

Type: **preferences <RETURN>**

This displays the usual preferences screen. Use your mouse to make sure that the printer settings are correct for your printer if you want to carry out the printing operations below. Save your preferences and you will return to the CLI prompt 1>.

5. **Print the directory**

Type: **dir >par: <RETURN>**

The system will send the directory to the parallel printer port, and it will be printed out. (I'm assuming that your printer has a parallel interface.) You can use the same command syntax later to get hard copy of directories in the Public Domain programs.

6. **Print the help.doc file**

Type: **type help.doc to par: <RETURN>**

The help file will be printed out. You can use the same command syntax later to get hard copy of the documentation files in the public domain programs.

7. **View the help.doc file**

Type: **type help.doc <RETURN>**

You will now see the help file on the monitor screen.

8. **Execute Move-Commands**

Type: **execute move-commands <RETURN>**

The screen will tell you that the RAM drive is being set up, various DOS commands copied, and you get some updated disk information. **Blobs** and **Dazzle** as examples.

9. **View and Print the Directory**

Type: **cd df0: <RETURN>**

The Amiga will now "recognise" and accept your public domain disk. To view, and/or print out the directory, use the same command syntax as in Steps 3 and 5 above.

Note the text files README.list15 and README.dist. These contain some general instructions and information, and may be printed and/or viewed by following the command syntax explained in Steps 6 and 7 above.

10. **Run the Library Programs**

First, the bad news! Some of these programs cannot be used because they are either too large for the system under a Ramdisk, or because they require AbasiC. The good news is that there are still many worthwhile programs that you **can** run. I will use **Blobs** and **Dazzle** as example programs. The AmigaLibDisk15 directory includes:

blobs (dir)
dazzle (dir)

These are sub-directories containing various associated programs. To get into the HblobsH directory:

```
Type:   cd blobs   <RETURN>
        dir        <RETURN>
```

You will now see the directory of the **Blobs** programs. To run **Blobs**:

Type: **blobs <RETURN>**

This program has a menu bar at the top left of your screen. You can access it with your mouse. When you feel like a program change, use the mouse to exit **Blobs**, and get back to the 1> prompt.

Suppose that you now want to run **Dazzle**. You will have to return to the root directory and then change into the **Dazzle** sub-directory:

```
Type:   cd df0:     <RETURN>  (change to root directory)
        dir         <RETURN>  (display root directory)
        cd dazzle   <RETURN>  (change to dazzle directory
        dir         <RETURN>  (display dazzle directory)
```

Now you will see the **dazzle** sub-directory. To run **Dazzle** use the same procedure that you used to run **Blobs**. Now your efforts will be rewarded by continuous spectacular graphics! Sit back and enjoy a well-deserved drink while you watch them.

New Public Domain Disk Volumes

AUG Library Disk 25

This disk contains a port (executables only) of the popular UNIX game "Hack", courtesy of John Toebes, 120 H Northington Pl, Cary NC. 27511.

This is Version 1.0.1E., with graphics enhancements for the Amiga.

AUG Library Disk 26

UnHunk Tool to process the Amiga "hunk" loadfile format. Collects code, data, and bss hunks together, allows individual specification of code, data, and bss origins, and generates binary file with format reminiscent of Unix "a.out" format. The output file can be easily processed by a separate program to produce Motorola "S-records" suitable for downloading to PROM programmer.

C-kermit Port of the popular "kermit" program, a flexible virtual terminal and file transfer program from Columbia University.

Ps A program to display process priorities and another program to set a process's priority to any arbitrary valid value. Very useful to provide finer control over your multitasking environment.

Archx Yet more programs for bundling up text files and mailing or posting them as a single file unit.

AUG Library Disk 27

ABdemos Amiga Basic demos. NewConvertFD creates .bmaps from fd files. BitPlanes finds addresses of and writes to bitplanes of the screen's bitmap. AboutBmaps is a tutorial on creation and use of bmaps. LoadILBM loads and displays IFF ILBM pics. LoadACBM loads and displays ACBM pics. ScreenPrint creates a demo screen and dumps it to a graphic printer.

Disassem A simple 68000 disassembler. Reads standard Amiga object files and disassembles the code sections. Data sections are dumped in hex. The actual disassember routines are set up to be callable from a user program so instructions in memory can be disassembled dynamically.

DvorakKeymap Example of a keymap structure for the Dvorak keyboard layout. Untested but included because assembly examples are few and far between.

Hypocycloids An electronic Spirograph inspired by an article in Feb '84 Byte.

LinesDemo A demo program which illustrates the use of proportional gadgets to scroll around in a superbitmap window.

MemExpansion Schematics and directions for building your own homebrew 1 Mb memory expansion.

SafeMalloc Ever have a program that corrupts it's own memory? Well, this little gem can help you find the problem in a hurry. Acts as an interface between your program and the real malloc, checking for overrun, underrun, and duplicate freeing of malloc'd space.

ScienceDemos Some science demos. Sidereal is a tutorial program which introduces the user to the relationships between Julian and calendar dates and solar and sidereal times. J2000 is a utility program which converts stellar positions, proper motion, parallax and radial velocity from the standard epoch B1950 (FK4) to epoch J2000 (FK5). Galilean is a tutorial program which determines the position of the Galilean satellites relative to Jupiter.

AUG Library Disk 28

Backgammon Another game in AbasiC by the author of Monopoly.

Cpp This is a copy of the Decus cpp, ported to the Amiga. This cpp is more powerful and complete than either of the built in cpp's in Manx or Lattice C. Also included is a modified version of the Unix like cc frontend, for Manx C, that knows about the cpp.

Cribbage Another game in AbasiC by the author of Monopoly.

MileStone Another game in AbasiC by the author of Monopoly.

Othello Another game in AbasiC by the author of Monopoly.

Shar A program which can pack and unpack archives compatible with the Unix "shar" (shell archiver) program.

SuperBitMap An example program that shows how to use ScrollLayer, how to sync the SuperBitMap prior to printing, and how to create a dummy rastport for dumping the SuperBitMap. Works under 1.2 and up.

AUG Library Disk 29

AegisDrawDemo Demo disk of the Aegis Draw program. This is the actual production program with only the "save" feature disabled and without documentation (intentionally). Very impressive program!

Cc Version of the Unix like C compiler frontend program, for Manx C. Previous released version was for Lattice C.

Enough A new CLI execute file command that tests for various system resources such as available memory or existance of specific files, directories, or devices.

Player Animation player for Aegis Animator.

Rubik An animated Rubik's cube program based on Barry Whitebook's "amiga3d" program and Raymond Brand's "skewb" program.

StringLib A public domain reimplementation of the Unix string library functions.

Vt100 A vt100 emulator program with Kermit and Xmodem file transfer protocols (based on AmigaTerm).

AUG Library Disk 30

Note that the programs on this disk are all shareware/freeware. This means that if you find them useful, you are morally obligated to send a small donation to the author to help support his efforts and fund further development. I hope that by including them in this library I have furthered the author's goals of widest possible distribution and thus maximum return.

Bbs A BBS for the Amiga (BBS-Amiga Version 1.1).

FineArt This is a collection of works from some of the best of current Amiga artists. Many thanks to Jim Sachs, Sheryl Knowles, Jack Haeger, and Aegis Development for submissions.

FontEditor Does what the name implies, edits fonts.

MenuEditor Create and edit menus, saving the result as either a binary file for further editing or as C source for inclusion in a program.

StarTerm Starterm version 3.0. Very nice telecommunications program.

AUG Library Disk 31

Life Executes the cellular automata game of LIFE in the blitter chip. Uses a 318 by 188 display and runs at 19.8 generations per second.

Mandelbrot Latest version of Robert French's mandelbrot program, version 3.00.

MxExample Example of mutual exclusion gadget handling.

RamSpeed Program to measure raw memory speed, comparing internal memory to the external memory.

Set Replacement for the Manx "set" command (to set or change environment variables) with several improvements.

Tree Draws a recursive tree (green leafy type, not files).

TxEd Latest demo version of Microsmith's text editor, TxEd. This is the full production version except that files are limited to 10K bytes in length and the search/replace functions are disabled. Also, demo is based on an older version of TxEd, new release has additional features.

VDraw Drawing program, based on freedraw, but now transformed and enhanced beyond recognition. Draws hollow boxes, filled boxes, freehand sketches, lines, circles, and arcs. Also has area fill, text insertion, 16 different line patterns, predefined area fill patterns, a magnify mode, cut and paste, color inversion, erase, grid pattern, and more. This is version 1.08.

Xicon Contains two programs; Xicon which lets you use icons to call up scripts containing CLI commands, and Ticon, which is a simple program to display text files from icons.

AUG Library Disk 32

Address Extended address book written in AmigaBasic.

Calendar Calendar/diary program written in AmigaBasic.

DosPlus1 First volume of CLI oriented tools for developers. Executables only.

DosPlus2 Second volume of CLI oriented tools for developers. Executables only.

MacView Allows viewing of a standard MacPaint picture file in either Amiga low resolution (320 x 200) or high resolution (640 x 400). Executable only.

Puzzle Simulation of puzzle with moving square tiles. Executable only.

ShowHAM Program to display HAM (Hold And Modify) mode images from the CLI environment. Executable only.

Solitaire Two new ABasiC games, Canfield and Klondike.

Spin3 Simple program that creates spinning cubes and transforms them into op-art. Example of how to create a double buffered display with color tables that can be changed.

Sword Sword of Fallen Angel. Text adventure game written in AmigaBasic.

Trails Cute little program that leaves a trail behind the pointer when the mouse is moved. The trail has programmable symmetry, thickness, and length. The other interesting thing about this program is that it is written in Modula-2 (source provided).

A Brief Look At The Latest Disks

If you're writing software in Lattice C or the Amiga Macro assembler, then get yourself a copy of **BLINK** from disk 34. Throw away ALINK, 'cos BLINK is faster and generates smaller object files. Since ALINK and BLINK are completely compatible, simply rename BLINK to ALINK and keep using your existing execute files.

Deluxe Paint users might like get disk 34, which contains **28 new font** files. If you're into video games, the **Triclops** program on disk 35 is certainly worth a look.

Want to show off your Amiga? Get disk 33, and run the **Sproing** program via the CLI run command. I won't explain what it does, just try it out. **Oing** does much the same thing, less sound. Guaranteed to upset Atari owners.

Some software companies are now putting demo versions of their programs into the public domain. Check out disk 29 for a demo version of **Aegis Draw**, and disk 31 for **Microsmith's TxEd.**

AUG Library Disk 33

3dstars 3d version of Leo's "stars" program (also on this disk).

Bigmap Program which demonstrates how to use the low-level graphics calls, ScrollVPort especially. Demonstrates scrolling around inside a very big bitmap.

Dbuf.gels Sample program that demonstrates the animation routines for Bobs and VSprites. Uses double buffering to smooth the display motion.

DiskMapper Displays sector allocation of floppy disks.

MemView Program which sets up a direct window into RAM, thus dynamically displaying the contents of memory.

Oing Displays a window full of little bouncing balls.

ScreenDump Dumps rastport of highest screen/window to printer.

Sdb Simple database program, originally released on a DECUS VAX SIG tape.

Sproing Same as Oing but includes sounds of balls colliding with boundries.

Stars Displays a screen full of stars, reminiscent of a view from the starship Enterprise's flight deck.

TermPlus Yet another variant of Michael Mournier's AmigaTerm program. This one includes improved ascii capture, CRC and checksum xmodem protocol transfers, CompuServe B-protocol transfers, a phone library, function key support, and limited AmigaDOS functions.

Vt100 Release 2.0 of Dave's version of AmigaTerm. Includes support for function keys and script files.

AUG Library Disk 34

Alint Support files for Gimpel lint to make it useful on the Amiga.

Blink A linker written as a replacement for Alink. Fully Alink compatible and supports many additional options not found in Alink. Also is much faster than Alink and generates smaller executable files. Version 5.7

Browser Updated version of the browser program released on disk number 18. This one has been "manxified", has scroll bars, and several bugs fixed.

Btree Routines to implement a B-tree algorithm and several accompanying tests, apparently derived from the single file version also included on this disk under "Btree2". Still buggy, but looks useful with a little work.

Btree2 Previous (original) version of btree routines.

Calendar Appointment calendar that lets you visually add and update appointments. Can also run in background and remind you 15 minutes before any scheduled event.

Less Like Unix "more", only better, with forward and backward scrolling, searching and positioning by percent of file and line number, etc. Very useful!

NewFonts Set of 28 new Amiga fonts.

Pr A background print utility with several options for letter quality compressed mode and line numbers. Also takes multiple files and wildcards.

Requester A file name requester that looks like the one used in Deluxe Paint, and a sample program for using the requester.

AUG Library Disk 35

ASendPacket Example program for sending multiple packets asynchronously to a dos handler, for those interested in implementing programs with asynchronous AmigaDos file I/O.

ConsoleWindow Example program for finding the intuition pointer to an AmigaDos CON: or RAW: window, so you can do WindowToFront, graphics, and other interesting things. (Requires AmigaDos 1.2)

DirUtil Nice little program to wander around directory tree using a windowing interface and performing various operations on files.

DirUtil2 Another variant of dirutil.

FileRequester A very nice file requester module for lattice C programs, along with a demo driver program.

MacView Allows viewing of standard MacPaint picture files in either Amiga low resolution (320 x 200) or high resolution (640 x 400). Executable, sample pictures, and icons (version on disk 32 did not include pictures).

Plop Short, simple, no-frills IFF reader program, and a sample picture made using a ray tracing algorithm.

PopCLI Provides a simple way of starting another CLI at any time without having to load workbench or exit whatever program you may be using. Also has a builtin screen saver mode that automatically blanks the Amiga console screen when there has been no input for a specified period of time.

QuickCopy Three versions of quickcopy, a nice little full disk copier. Two of the versions are capable of making backup copies of "protected" Electronic Art's disks.

ScrollPf Creates and displays a 400 by 300 by 2 bit plane playfield on top of a 320 by 200, 2 plane deep playfield, as a demo of dual playfield display.

SendPacket General purpose subroutine to send AmigaDos packets.

SpriteMaker Program which lets you paint sprites and then converts the image into a C data structure to be used in programs. Allows interactive testing of the sprite as a pointer. Suggested shareware donation of $15 for source code and updates.

Tracker Program which converts a boot-load disk (I.E. a kickstart disk) into a group of files for electronic transmission and reassembly. Completely preserves the original disk structure so the target disk will be an exact duplicate of the original.

TriClops Very nice graphics oriented 3-D space invasion game. This was previously a commercial product which is being released into the public domain for promotional purposes (they are working on a multiuser, multimachine version).

Tsize A simple utility to print directory tree sizes. Displays the total size of all files and subdirectories within a given directory.

UnIfdef Useful program for removing ifdef'ed sections from a file while otherwise leaving the file alone. Allows one source to be used as a porting base for many machines, without shipping the entire source to every source customer. (Preprocess for their machine and send them only the source for their specific machine).

Vttest Program to test compatibility of vt100-compatible terminals and terminal emulators. Requires the resources of a Unix system to test an Amiga hosted vt100 emulator. (I haven't yet found one that even comes close to passing this test!).

If you are a **StarTerm** user, there is a new version on disk 30. Many more bells and whistles, but unfortunately no way to save your favourite settings. **vt100** is my current favourite terminal program, with both XMODEM and KERMIT file transfer protocols. Get the one on disk 33, not the older version from disk 29.

Using Diskfonts in Amiga Basic

Amiga Basic has many commands which allow you to take advantage of most of the advanced features of the machine, however there are some features which are not directly supported by specific commands. These problems can usually be overcome by using the LIBRARY command to access system routines which will do those things for you.

One of the things that it would be nice to do is to use all those fonts available on the Workbench disk to make your Basic programs look more exciting. You can look at what fonts are available by using the notepad utility on the workbench. There is no direct way of using the fonts but by accessing system routines you can use all of them.

There are two libraries which you need to open to use the disk fonts. These are the graphics library and the disk font library respectively. To use a library from Basic you need a .bmap file on the disk to tell the computer where to find the library routines and what they are called. There is a file in the basicdemos drawer called "graphics.bmap" but unfortunately there is no corresponding "diskfont.bmap". The format for creating a .bmap file is in the Basic manual but unfortunately the information that goes in the file is not supplied. There is also a program for converting an .fd file to a .bmap file but unfortunately again it doesn't describe what an .fd file is and there are none around to be found!

I found this to be an insoluble problem until I got my ROM Kernal Manual which had the information listed that needed to go into the .bmap file. To cut a long story short, the program below will create a "diskfont.bmap" file on the disk for you using information gleaned from the ROM Kernal Manual.

```
OPEN "DiskFont.bmap" FOR OUTPUT AS 1

readnext:
  READ a$
  IF a$ = "end" THEN CLOSE 1:END
  FOR z = 1 TO LEN(a$)
    PRINT #1,MID$(a$,z,1);
  NEXT z
  PRINT #1,CHR$(0);
  READ offsetup
  PRINT #1,CHR$(offsetup);
  READ offsetlw
  PRINT #1,CHR$(offsetlw);

reg = 1:WHILE reg <> 0
  READ reg
  PRINT #1,CHR$(reg);
WEND
PRINT#1,CHR$(0);:GOTO readnext

DATA "OpenDiskFont",&hff,&he2,9,0
DATA "AvailFonts",&hff,&hdc,9,1,2,0
DATA "end"
```

Once this file is on the disk, all that remains is to use the library routines in your program. The routines that we need to use are:

```
                OpenDiskFont()
                SetFont()
        and     CloseFont()
```

OpenDiskFont() is a routine in the diskfont library and only requires one parameter. This is the address in memory of a long integer array that has in it certain text attributes. The first element of the array has in it the address of a string which is the name of the font that you want. This string must be finished with an ASCII zero. The other element of the array holds information about the height of the font, its style, and whether or not it is on the disk or in memory. For instance, if the array name was texattr&() then the first element could be given by:

```
textattr&(0)=SADD("opal.font"+CHR$(0))
```

where the function SADD gives the actual memory location of the string and the string is terminated with a zero byte.

Once the array has been set up and the function OpenDiskFont has been called the font that is specified will be loaded into memory and a pointer to the position in memory of the font is returned. If your workbench disk is not in a drive then you will be prompted to insert it so that it can read the font off the disk. However the font is not actually used until the SetFont() function is called.

The SetFont() routine actually attaches a font to a window for use. The parameters for this routine are the RastPort address which is given by the function WINDOW(8), and the pointer that is returned by the OpenDiskFont() function.

Now whenever you print any text to the window that you specified it will be printed in the new font. This is difficult to explain but easy to do and I hope that the program below will make things clearer. I have written the main part of the program as a sub program so that you can use it in your programs with relative ease. I suggest that you call the function at the beginning of your program because it will prompt you to put the Workbench disk back in the drive. The function actually does the work of loading the font into memory and then later in the program you can Set the font.

```
LIBRARY "graphics.library"      'These instructions cause both
LIBRARY "diskfont.library"      'libraries to be opened.

DECLARE FUNCTION OpenDiskFont&() LIBRARY

CALL font("opal.font",11&,1&,0&)          'call the sub program.
LOCATE 8,10
CALL SetFont(WINDOW(8),fontptr&)          'set the font in the
                                          'output window.
PRINT"Opal Font - 11 points high."

STOP


SUB font(fontname$,height&,style&,mode&) STATIC
SHARED fontptr&
'*
'*fontname = name of font to look for on the disk, in the format
'*             name.font ie opal.font or sapphire.font
'* height = height of font,ie some fonts have more than one
'*            height.
'* style = italic/bold/underlined or a combination.
'*          plain = 0,italic = 1,bold = 2,underlined = 4
'*          Can also be changed using the SetSoftStyle routine as
'*          demonstrated in the libraries demo on the extras disk.
'* mode = romfont or diskfont(0 or 1)
'* The OpenDiskFont routine will look for the closest font which
'* which matches your specification.
'* fontptr = a pointer to the font after it has been loaded into
'*             memory - this pointer is a SHARED with the main
'*             program and thus can be used in the main program to
'*             Set the font.
'*
textattr&(0) = SADD(fontname$+CHR$(0))          'Set up attribute
textattr&(1) = height&*65536&+style&*256+mode&  'array.
rp&=WINDOW(8)

fontptr& = OpenDiskFont&(VARPTR(textattr&(0)))  'open the font.

IF fontptr&=0 THEN PRINT"Error - couldn't open font.":STOP

END SUB
```

If there are any questions about this or difficulties getting it working in your program then I would be happy to help. I will be at the next meeting or else you can ring me on 489-1584 during reasonable hours.

Using Operating System Graphics Routines in Amiga BASIC

The Microsoft basic supplied with the Amiga is a fully featured Basic and has many commands which enable you to use the features of the computer with relative ease. There are, however, many of the graphics routines of the computer which are not directly available as basic commands but which can be very useful to produce effects otherwise unavailable to you. This article attempts to introduce some of these routines and explain how to use them.

The LIBRARY statement

The computer keeps related sets of routines together in libraries so that they are all in one place and easy to use. One such set of routines is the graphics library which is actually part of the information loaded from the kickstart disk and is in memory all the time. Other libraries are kept on disk and must be loaded from the disk when needed. These are found in the LIBS: directory of the Workbench disk. To use the graphics routines in the graphics library you must first tell the computer which library you want to use. To do this you use the LIBRARY statement which in effect links the library to your program for use. In the case of the graphics library you use the statement :-

```
LIBRARY "graphics.library"
```

This statement looks on the disk for a file named graphics.bmap which on the Extras disk is found in the basicdemos drawer. This file tells BASIC where to find all the routines and it must be present in the drawer that you are working from, (in more technical terms it must be present in the current directory), or your program won't be able to find it and will give an error. To get around this you can either keep your program in the basicdemos drawer, put a copy of the graphics.bmap file in the drawer that you're working with, or use the statement CHDIR "basicdemos" at the start of your program.

Now you are ready to use the graphics library routines. There are a large number of them and I will only cover a few here that are relatively easy to use and useful.

Move&(Rp&,x&,y&)

In Basic the text cursor can be placed at any character position on the screen using the LOCATE x,y command. However with the Move() routine you can place text at any point on the screen and get it aligned perfectly with your graphics. You can even overlap text. Of course this routine is also useful for setting the graphics cursor to a particular place on the screen before executing other graphics commands.

To use this routine you must supply three parameters. The first of these need not worry you too much - it is a pointer to a rast port for the particular window that you are working with and is given by the function WINDOW(8). Therefore all you need to do is include in your program a line of code such as Rp& = WINDOW(8) and use this value. You don't have to know what it means. The other two parameters are simply the x and y pixel co-ordinates where you want the graphics or text to be drawn. Note that these values must be long integer values or you will get an error.

The short example below demonstrates this routine.

```
LIBRARY "graphics.library"  'open the library

Rp&=WINDOW(8)               'get the rastport pointer

a$="graphics routine move demonstration"
x&=100:y&=80:inc=4*3.14159/35:p = 0
FOR z = 1 TO LEN(a$)          'loop around, setting
  CALL Move&(Rp&,x&,y&)       'each character position
  PRINT MID$(a$,z,1);         'and printing them one
  x&=x&+8:y&=y&+3*SIN(p)      'at a time.
  p = p+inc                   '
NEXT z
looptillmouse: if mouse(0)=0 then looptillmouse
```

SetDrMd&(Rp&,Mode&)

This routine sets the drawing mode of which there are four! The default mode which is used most of the time is called JAM2 mode. In this mode whenever anything is drawn, first the background colour is drawn and then the foreground colour. This is especially noticeable when printing text over graphics because a whole block is drawn around each character. To get around this you can change the mode to JAM1 mode which as you may have guessed draws only the foreground color and won't disturb graphics underneath the text.

The other two modes are known as COMPLEMENT and INVERSVID and perform the functions of their names. COMPLEMENT mode does not use the pen colors for drawing but instead, the color of each screen dot where a pen was supposed to draw, is complemented. In binary, this means that all of the bits in the original color are inverted, ones becoming zeroes and visa versa. The resulting colour can easily be calculated for those not familiar with binary by taking the highest possible colour number and subtracting the current colour number. Thus if you are drawing on a background that is color 0 with a screen that has only 4 colors (0 - 3), the complemented color will be 3.

INVERSVID reverses the role of the background color and foreground colour but produces different effects with JAM1 and JAM2.

These different modes can only be set with the function SetDrMd() and the parameters are as follows. Rp& is exactly the same as for the Move routine and is equal to WINDOW(8). The Mode is a long integer equal to the desired modes.

```
JAM1 = 0; JAM2 = 1; COMPLEMENT = 2; INVERSVID = 4
```

These numbers can be added together to produce combinations of modes although obviously JAM1 and JAM2 are mutually exclusive. The following short program demonstrates the difference between these modes and it will probably be easier to understand the differences between the drawing modes once you have run it and have played around with them.

```
LIBRARY "graphics.library"
rp&=WINDOW(8)
JAM1& = 0
JAM2& = 1
COMPLEMENT& = 2
INVERSVID& = 4
LINE (100,0)-(300,60),2,bf
CALL SetDrMd&(rp&,JAM2&)
PRINT "This is JAM2 mode, the default mode."
PRINT "Note that the background is filled in as well."
CALL SetDrMd&(rp&,JAM1&)
PRINT "This is JAM1 mode. Note that the background"
PRINT "is not changed."
CALL SetDrMd&(rp&,JAM1&+COMPLEMENT&)
PRINT "This is COMPLEMENT mode."
CALL SetDrMd&(rp&,JAM2&+INVERSVID&)
PRINT "This is INVERSVID mode where the pen and background"
PRINT "colors are reversed."
CALL SetDrMd&(rp&,JAM2&)
COLOR 1,0
```

PolyDraw(Rp&,numpoints&,arrayptr&)

PolyDraw is another useful routine in that it will accept an integer array of points and then plot lines between them. This is similar to the basic command AREAFILL except that it doesn't join the end point to the start point and it doesn't do any fill operation. It is very fast for drawing polygons or more complex pictures and is much faster than using a series of LINE commands.

To use PolyDraw you must first fill an integer array with the x and y coordinates of the points that you want to draw between. The routine uses the current pixel cursor location as its starting point and then draws to the first point specified in the array. Thus unless you want to start drawing from 0,0 you must first set the pixel location with the Move() routine.

The first parameter to pass is as for the other routines. The second parameter ,numpoints, is the number of coordinate pairs or points that you want to draw. This will always be half the size of the array that you have set up. The coordinate points are placed in the single dimensional array in the order that you want them drawn, with x cordinate followed by y coordinate. For instance if the array is called p% and the points are

x…xn and y…yn (n = no. of points)

then the array will be

p%(0)=x,p%(1)=y,p%(2)=x1,p%(3)=y1…etc to n.

The actual parameter passed to the routine is not the array name itself but the address in memory of that array. This is given by the function VARPTR().

The following short program illustrates the use of this routine.

```
LIBRARY "graphics.library"

SCREEN 2,320,200,4,1
WINDOW 2,"stars",,15,2

DEFINT x,y,z
DIM points%(19) '10 co-ordinate pairs
FOR z = 0 TO 19                 'loop to read in all
  READ points%(z)               '10 points into the array.
NEXT
  DATA   108,112,130,115,113,125,121,145,100,133
  DATA   79,145,87,125,70,115,92,112,100,90

rp&=WINDOW(8)

x=100:y=90
col = 11
FOR c = 0 TO 4                                   'draw 5 times
  COLOR col
  CALL Move&(rp&,x,y)                            'set start point
  CALL PolyDraw(rp&,10,VARPTR(points%(0)))       'draw it
  FOR z = 0 TO 19 STEP 2
    points%(z)=points%(z)+20                     'move x across 20
    points%(z+1)=points%(z+1)-15                 'move y up 15
  NEXT z
  x = x+20:y = y-15:col = col+1    'move start point and change
NEXT c                             'colour.

WHILE MOUSE(0)=0
  FOR col = 11 TO 15           'produce a pretty
    PALETTE col,RND,RND,RND    'effect.
  NEXT
WEND

END
```

There are a number of other graphics routines that may be quite useful however there is not space to go into them here. They are all explained in the Amiga Rom Kernel Manual if you want to know more.

As was mentioned in the beginning there are other libraries apart from the graphics library but there are no associated .bmap files for them on the Extras disk. Microsoft has kindly provided a program for converting .fd files to .bmap files however the .fd files are also missing!! This can be rectified and in further articles I will explain how to do it.

Editor's Column

Hi again, and welcome to our 5th issue of the club newsletter. After last month's issue, which several wags called a reprint due to the severe lack of original articles, I'm most happy to see that **all** the articles in this issue are locally written. I'd like to thank the members who have taken the time to write for the newsletter, and plead with those who haven't to put pen to paper (or fingers to keyboard!). By the way, if you send your articles on disk, PLEASE write your name and address on the disk label, and remember to put your name at the beginning or the end of the article. If you can possibly arrange it, save the text in **pure ascii** format.

The major news this month, apart from the 11 new public domain disks, is that we've managed to find a larger room to hold our meetings in at Victoria College, and it's only a few metres from our present meeting rooms. Lecture Theatre 2 seats 200 people, compared with the 75 seats we've put up with so far. For the time being, we've hired both the new room **and** the old rooms, because the lecture theatre is really only suitable for holding the main meeting, and not for socialising. We'll have to play things by ear until we're properly sorted out.

Unfortunately, the new meeting place also means that we'll be paying $100 per meeting instead of the $40 we've been paying so far. To make up the extra money, we'll need to sign-up more Amiga owners. Wherever you go, we expect you to tell everyone the advantages of being a member of the Amiga Users Group. In my estimation, there have been about 800 to 1000 Amigas sold in Melbourne. The Amiga Users Group represents about 220 of these people, but we'd like to have **all** of them as members. If you meet an Amiga owner who isn't yet a member, get him or her to fill out the application form at the back of the newsletter and send it in. Consider yourself a recruitment officer, and make it your job to sign up at least one new member each.

Club Meetings

Club meetings are held at 2pm on the second Sunday of each month at Victoria College, Burwood Campus, in Lecture Theatre 2, and the adjacent Community Resources Centre, Building E.

The dates of our next few meetings are **October 12th, November 9th** and **December 14th.**

AUG Users Group Disks

Disks from the **AMIGA Users Group** Library are available on quality 3.5" disks for $10 each including postage. The group currently holds 35 public domain volumes, sourced from the USA.

Member's Discounts

The **AMIGA Users Group** is currently negotiating discounts for its members on hardware, software and books. Members will be notified when negotiations are complete.

Currently, **Technical Books** in Swanston Street in the city offers **AUG** members a 10% discount on computer related books. Just show your membership card.

Your Committee

John Holland . . .	(Co-ordinator)	348 1358
Bob Scarfe	(Vice Co-ordinator)	376 4143
Ron Wail	(Meeting chairman)	878 8428
Eric Salter	(Secretary)	861 9117
Paul Radford . . .	(Treasurer)	663 3951 BH
Neil Murray	(Membership)	792 9666
Stephen Thomas . .	(Membership)	830 5783
Peter Jetson . . .	(Newsletter editor)	762 1386
Geoff Sheil	(Assoc editor)	509 3151
Jo Santamaria . . .	(Assoc editor)	836 9129
Ron Van Schyndel .	(Librarian- books)	882 7264
Geoff Wood	(Librarian- books)	580 7463
Ron Wail	(Librarian- software) . . .	859 2998
Mike Creek	(Librarian- software) . . .	
Bohdan Ferens . . .	(Purchasing)	792 1138
Justin Somers . . .	(Without portfolio)	553 0379

Membership and Subscriptions

Membership of the **AMIGA Users Group** is available for an annual fee of $20. To become a member of AUG, fill in the form on this page (or a photocopy of it), and send it with a cheque for $20 to:

AMIGA Users Group, PO Box 109, North Balwyn, 3104

SOFTWARE ORDER FORM

Disk numbers :

Disks supplied by Amiga User Group @ $10	$
Disks supplied by member @ $2	$
Club Use Only Receipt #: Mailed on: / /	Total $

Mail to: Amiga Users' Group, P.O. Box 109, North Balwyn, 3104.

Member's Name:

Address:

Application for membership of The Amiga Users Group

Membership is $20 per year. Make cheques payable to The Amiga Users Group, and send to:

Amiga Users Group, PO Box 109, North Balwyn, 3104

Surname: ______________________

First name: ______________________ (no initials)

Address: ______________________

______________________ Postcode: __________

Where did you hear about AUG: ______________________

How can AUG help you: ______________________

____

____

Signed: ______________________ Date: __________

In the event of my admission as a member, I agree to abide by the rules of the Association for the time being in force.

Details this side are optional

Phone number: ______________ STD code: ______

Year of birth: ______________________

Occupation: ______________________

Interests: ______________________

____

____

Dealer's Name: ______________________

Dealer's Address: ______________________

______________________ Postcode: __________

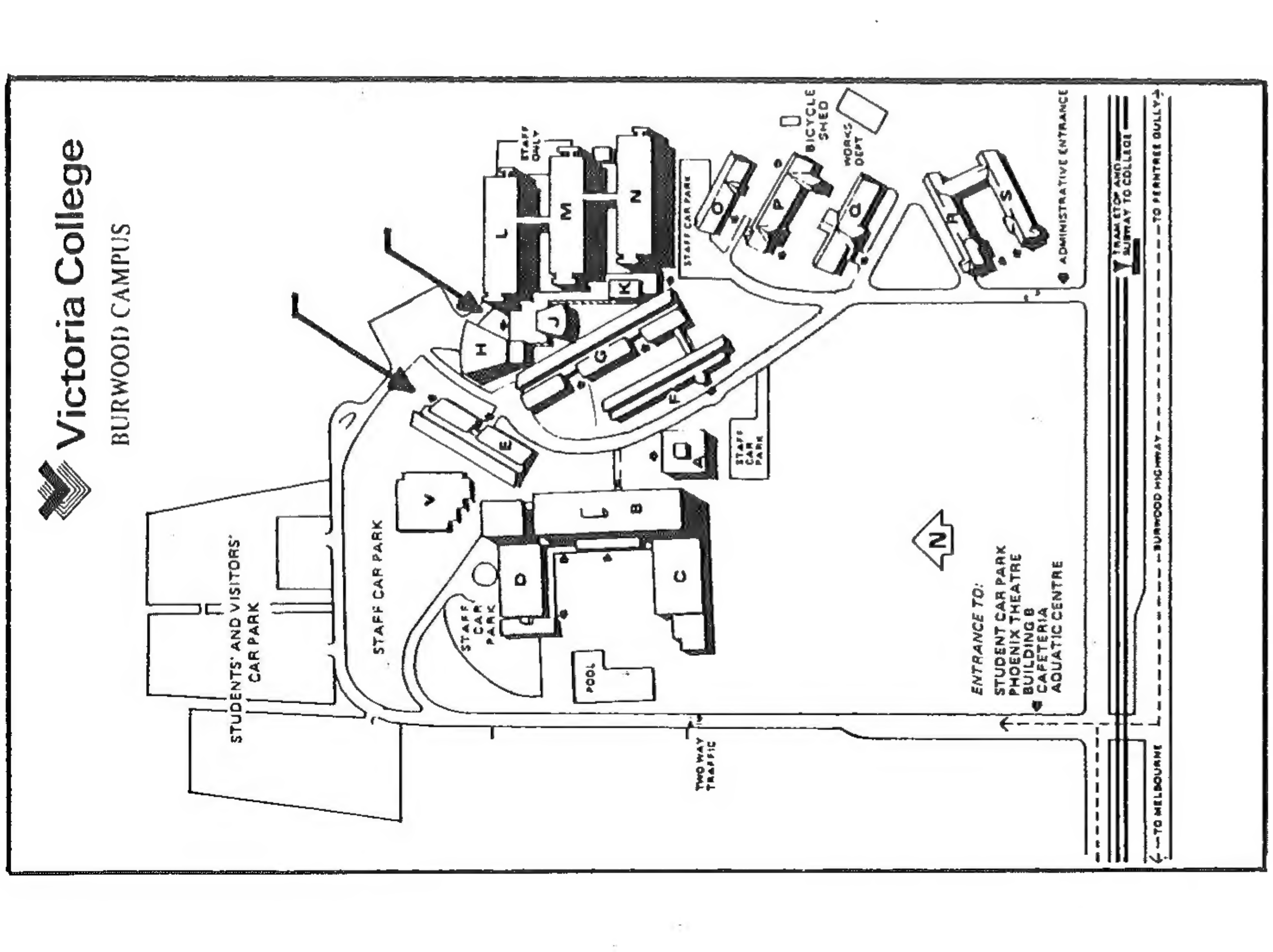
Are you happy with your dealer: ______________________

AMIGA Users Group
PO Box 109, North Balwyn, 3104
Victoria, Australia

AMIGA™ Users Group

P.O. Box 109, North Balwyn, Victoria, Australia, 3104

October 1986 Amiga Workbench



Where is Victoria College Burwood Campus?

New members and visitors sometimes have trouble locating our meeting place the first time. Victoria College is on the North side of Burwood Highway, Burwood, Just East of Elgar Road. Coming from the City, turn left at the first set of traffic lights after Elgar Road. Follow the road around past the football oval, over three or four traffic bumps to the car parking areas near the netball courts. Further up the road, you'll find building E and to the left, you'll find the location of our main meetings, Lecture Theatre 2.

For those with a Melways, try Map 61, reference B5. There should be a map of Victoria College on the back of the newsletter, how readable it is will depend on how much magic our printer was able to conjure up.